Chanalyzer: New Release

Abstract

Here is the description of the improvements and updates in the new Chanalyzer release.

Changes can be roughly classified into three main contributions:

1. the introduction of a new step (Step E) aimed at providing new metrics for the retrieved channel and new visual representations of it;
2. a modified pipeline (including new steps and programs) in case a pathway is provided as input.

1 New Step: E - Metrics and Visualization

We have integrated in Chanalyzer some metrics for evaluating, analyzing, and visually representing the channel retrieved by the previous steps (A - D2). The tools and the code have been introduced in this repository https://github.com/rea1991/GEO-Nav-methods and discussed in this article [RFB23].

Specifically, Step E takes as input the centerline endowed with radius values of the maximal inscribed balls returned by Step D2 and it provides as output of Step E:

- metrics about the centerline of the identified channel, such as:
  - the number of vertices,
  - the length,
  - the straightness;
- informative visualizations of the channel:
  - the centerline of the retrieved channel colored in accordance with the radius values adopting the coolwarm colormap of Matplotlib (represented as a point cloud and encoded into an OFF file),
  - the retrieved channel (as a collection of spheres) colored in accordance with the radius values adopting the coolwarm colormap of Matplotlib (represented as a point cloud and encoded into an OFF file),
  - a PNG file representing the graph of the radius functions of the centerline of the considered model.

Please notice that Step E can be applied to a channel returned by Chanalyzer independently by the fact that the adopted pipeline is the standard one proposed in the original Chanalyzer release or any of the modified ones which will be presented in the following.

2 Modified Pipeline in case of Input Pathway

We have experienced that for certain molecular structures, the portion of surface returned by Chanalyzer includes the correct channel but it also contains large portions of the molecular surface which do not have such property. A possible way to improve the result in such a situation consists in including information about the pathway and exploiting it to correctly identify the portion of the Chanalyzer output being a channel.

Here, the proposed pipeline to face a similar scenario (please read carefully the documentation of each step and consider the example included in the corresponding folders):

- Perform Step A as in the standard pipeline (actually, the only output file needed to the next steps is the triangulated surface encoded in `triangulatedSurf.off`; if such a representation is already available, this step can be skipped);
- Retrieve the channel by running the program `B_pathway_channel_projection.py` taking as input the triangulated surface obtained at the previous step, the provided pathway (represented via a .xyz file consisting of a collection of lines each of which encodes the coordinates of a pathway point);
- Perform Step C1 as the standard pipeline in order to obtain the skeleton of the channel;
- Process the skeleton by running the program `C2_pathway_skeleton_processing.py` (a slight modification of Step C2) that returns a path from the two channel entrances;
- Perform Step D1, Step D2, and Step E as in the standard pipeline (in the scenario in which a pipeline is provided, we experienced that the centerline produced by Step D1 may lay outside the channel; in such a case, skip Step D1 and take the path produced by Step C2_pathway.)

3 Minor updates

We have:

- corrected some typos and modified the NanoShaper Installation Guide;
- added a python program `pqr2xyzr.py` to convert structures expressed in a 'pqr' format into a 'xyzr' format.

References

[RFB23] Andrea Raffo, Ulderico Fugacci, and Silvia Biasotti. Geo-nav: A geometric dataset of voltage-gated sodium channels. *Computers & Graphics*, 115:285–295, 2023.